

- [67] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. C. Rinard, “Managing Performance vs. Accuracy Trade-offs With Loop Perforation,” in *SIGSOFT/FSE’11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and ESEC’11: 13th European Software Engineering Conference (ESEC-13)*, pp. 124–134, 2011.
- [68] Avizienis and Kelly, “Fault Tolerance by Design Diversity: Concepts and Experiments,” *Computer*, vol. 17, no. 8, pp. 67–80, 1984.
- [69] T. Y. Chen, F. Kuo, R. G. Merkel, and T. H. Tse, “Adaptive Random Testing: The ART of test case diversity,” *J. Syst. Softw.*, vol. 83, no. 1, pp. 60–66, 2010.
- [70] T. Jackson, *On the Design, Implications, and Effects of Implementing Software Diversity for Security*. PhD thesis, University of California, Irvine, 2012.
- [71] G. R. Lundquist, V. Mohan, and K. W. Hamlen, “Searching for Software Diversity: Attaining Artificial Diversity through Program Synthesis,” in *Proceedings of the 2016 New Security Paradigms Workshop, NSPW ’16*, (New York, NY, USA), p. 80–91, Association for Computing Machinery, 2016.
- [72] B. Randell, “System Structure for Software Fault Tolerance,” *SIGPLAN Not.*, vol. 10, p. 437–449, apr 1975.
- [73] N. Harrand, *Software Diversity for Third-Party Dependencies*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2022.
- [74] J. V. Cleemput, B. Coppens, and B. D. Sutter, “Compiler Mitigations for Time Attacks on Modern X86 Processors,” *ACM Trans. Archit. Code Optim.*, vol. 8, no. 4, pp. 23:1–23:20, 2012.
- [75] A. Homescu, S. Neisius, P. Larsen, S. Brunthaler, and M. Franz, “Profile-guided Automated Software Diversity,” in *Proceedings of the 2013 IEEE/ACM International Symposium on Code Generation and Optimization, CGO 2013, Shenzhen, China, February 23-27, 2013*, pp. 23:1–23:11, IEEE Computer Society, 2013.
- [76] S. Bhatkar, D. C. DuVarney, and R. Sekar, “Address Obfuscation: An Efficient Approach to Combat a Broad Range of Memory Error Exploits,” in *Proceedings of the USENIX Security Symposium*, 2003.
- [77] S. Bhatkar and D. C. DuVarney, “Efficient Techniques for Comprehensive Protection from Memory Error Exploits,” in *Proceedings of the 14th USENIX*, 2005.
- [78] K. Pettis and R. C. Hansen, “Profile Guided Code Positioning,” in *Proceedings of the ACM SIGPLAN’90 Conference on Programming Language Design and Implementation (PLDI)*, pp. 16–27, 1990.

- [79] S. Crane, A. Homescu, S. Brunthaler, P. Larsen, and M. Franz, “Thwarting Cache Side-channel Attacks Through Dynamic Software Diversity,” in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*, The Internet Society, 2015.
- [80] A. Romano, D. Lehmann, M. Pradel, and W. Wang, “Wobfuscator: Obfuscating JavaScript Malware via Opportunistic Translation to WebAssembly,” in *2022 IEEE Symposium on Security and Privacy (SP) (SP)*, (Los Alamitos, CA, USA), pp. 1101–1116, IEEE Computer Society, may 2022.
- [81] M. T. Aga and T. M. Austin, “Smokestack: Thwarting DOP Attacks with Runtime Stack Layout Randomization,” in *IEEE/ACM International Symposium on Code Generation and Optimization, CGO*, pp. 26–36, 2019.
- [82] S. Lee, H. Kang, J. Jang, and B. B. Kang, “SaVioR: Thwarting Stack-based Memory Safety Violations by Randomizing Stack Layout,” *IEEE Trans. Dependable Secur. Comput.*, vol. 19, no. 4, pp. 2559–2575, 2022.
- [83] Y. Younan, D. Pozza, F. Piessens, and W. Joosen, “Extended Protection against Stack Smashing Attacks without Performance Loss,” in *22nd Annual Computer Security Applications Conference (ACSAC 2006), 11-15 December 2006, Miami Beach, Florida, USA*, pp. 429–438, IEEE Computer Society, 2006.
- [84] Y. Xu, Y. Solihin, and X. Shen, “MERR: Improving Security of Persistent Memory Objects via Efficient Memory Exposure Reduction and Randomization,” in *ASPLOS ’20: Architectural Support for Programming Languages and Operating Systems*, pp. 987–1000, 2020.
- [85] G. S. Kc, A. D. Keromytis, and V. Prevelakis, “Countering Code-injection Attacks With Instruction-set Randomization,” in *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS*, pp. 272–280, 2003.
- [86] E. G. Barrantes, D. H. Ackley, T. S. Palmer, D. Stefanovic, and D. D. Zovi, “Randomized Instruction Set Emulation to Disrupt Binary Code Injection Attacks,” in *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS*, pp. 281–289, 2003.
- [87] M. Chew and D. Song, “Mitigating Buffer Overflows by Operating System Randomization,” Tech. Rep. CS-02-197, Carnegie Mellon University, 2002.
- [88] D. Couroussé, T. Barry, B. Robisson, P. Jaillon, O. Potin, and J. Lanet, “Runtime Code Polymorphism as a Protection Against Side Channel Attacks,” in *Proceedings of Information Security Theory and Practice - 10th IFIP WG 11.2 International Conference, WISTP*, vol. 9895, pp. 136–152, 2016.